EXHIBIT 1

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE	
Declaration Under 37 C.F.R. 1.131	Atty. Docket No. **VIGN1370-1**

Applicants **Eric R. White**	
Application Number **10/036,980**	Date Filed **12/31/2001**
Title **System and Method for Providing a Public Application Program Interface**	
Group Art Unit **2194**	Examiner **Wu, Qing Yuan**
Confirmation Number: **5326**	

Certificate of Mailing Under 37 C.F.R. §1.10

I hereby certify that this document is being deposited with the United States Postal Service as Express Mail No. EV704313533US in an envelope addressed to: Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22312-1450 on 10-14-05

Julie H. Blackard

1. I, Eric R. White, am an original joint inventor of the invention described and claimed in the above-referenced patent application.

2. The invention claimed in the above-referenced patent application was conceived at least as early as July 27, 2001.

3. I drafted and sent the email attached as Exhibit A at least as early July 27, 2001.

4. To the email of July 27, 2001 I attached a Functional Specification drawn up by myself on or prior to July 27, 2001. The Functional Specification attached to the email of July 27, 2001 (Exhibit A) is attached hereto as Exhibit B.

5. The Functional Specification attached hereto as Exhibit B demonstrates a conception of the invention described and claimed in the above-referenced patent application at least as early as the date of the email: July 27, 2001.

6. Moreover, the invention claimed in the above-referenced patent application was reduced to practice as a prototype program at least as early as July 27, 2001. I drafted and sent the email attached as Exhibit C on July 30, 2001. The email attached hereto as Exhibit C was originally sent to Vignette's patent counsel Gray Cary Wary and Friedenrich, LLP and notes that a reduction to practice of the invention described and claimed in the above-referenced patent application prior to the date of the email: July 30, 2001. Specifically, in the July 30 email I note "we're working on a much more detailed description of the Extensible Workflow Architecture. We have a lot of knowledge about this one, *because we've built it"* (emphasis added).

8. Declarant acknowledges that willful false statements and the like are punishable by fine or imprisonment, or both (18 U.S.C. 1001) and may jeopardize the validity of the application or any patent issuing thereon.

I, Eric R. White, aver that all statements made of my own knowledge are true and all statements made on information and belief are believed to be true.

Eric R. White
September 30, 2005

EXHIBIT A

Diana Schaffer

From: Pastrana, Armando
Sent: Friday, July 27, 2001 10:07 AM
To: Adair, John
Subject: FW: provisional patent text

Attachments: ExtensibleWfConcepts.doc

ExtensibleWfConce pts.doc (33 K...

FYI

>. Armando Pastrana, Jr.
> GRAY CARY. TECHNOLOGY'S LEGAL EDGE
> 1221 South MoPac Expressway, Suite 400 Austin, TX 78746-6875
> 512-457-7080 (direct) 512-784-7080 (cell)
> 512-457-7001 (fax)
> Assistant: Katherin Cope (512) 457-7057
>

-----Original Message-----
From: White, Eric [mailto:ewhite@vignette.com]
Sent: Friday, July 27, 2001 09:26 AM
To: 'Pastrana, Armando'
Cc: White, Eric
Subject: provisional patent text

Armando,
I'm attaching a portion of a Functional Specification that contains some of the concepts surrounding the extensible Workflow architecture. I'll bulletize specific points below:

1. Extensible Workflow Architecture

The goal is to develop a single, standards-based interface to multiple, disparate workflow systems. Furthermore, the set of workflow systems supported should be easily extensible in the field; i.e., not requiring a rewrite or modification to the base source code for the architecture.

a) Intended to abstract vendor-specific implementations and interfaces of workflow engines and present a single, standards-based public API to workflow client applications.
b) Intended to provide a simple mechanism for extending the set of workflow engines, plugged in to this architecture via adaptors.
c) Intended to abstract vendor-specific syntax for documenting workflow definitions by adopting the Workflow Management Coalition (WfMC) XML Process Definition Language (XPDL) as the basis for describing workflow definitions.
XPDL is the translated, within the respective adaptor, to the target vendor-specific syntax for runtime execution.
d) There will be a configuration step to select one or more workflow engines to support under this architecture.
e) There will be support for multiple, simultaneous workflow engines under this architecture; i.e., you can configure the runtime system to direct API calls to multiple workflow engines running in parallel -- these engines don't have to be from the same vendor.
f) At the client API level, this architecture supports multiple, simultaneous client connections and presents a singular authentication and authorization interface for ensuring appropriate access by these client applications.
g) This architecture also supports distributed workflow insofar as the

following:
*) workflow activity implementations do not have to co-reside with the workflow engine
*) workflow clients may connect from remote machines
*) other, distributed workflow engines may interact with an engine wrapped in the extensible architecture via subflow calls. This subflow interface is also governed by standards.

2. Transition Syntax

This initiative addresses a missing portion of the WfMC specification for XPDL (http://www.wfmc.org/standards/docs/xpdl_010522..pdf). In XPDL the syntax for workflow activity transitions is undefined, rendering this specification worthless in abstracting vendor-specific syntax for these expressions.

The approach to solving this problem is to create an XML Document Type Definition (DTD) to govern XML syntax for expressing workflow activity transitions. Use of XML permits easy parsing and translation to vendor-specific syntax when the transition syntax needs to be employed in a vendor-specific engine.

The base of the XML DTD is the XEXPR Scripting language proposal (http://www.w3.org/TR/xexpr/) of the World-Wide Web Consortium. XEXPR is not completely suitable for direct adoption and, so, must be customized for use as a standard in workflow transitions expression syntax.

Workflow transition expressions must all result in boolean (true, false) values, since they participate in the decision logic for traversing paths between activities in a workflow process definition.

Furthermore, workflow transition expressions must support the following data types as participants in boolean expressions:

1) Integer
2) Float
3) String
4) Boolean

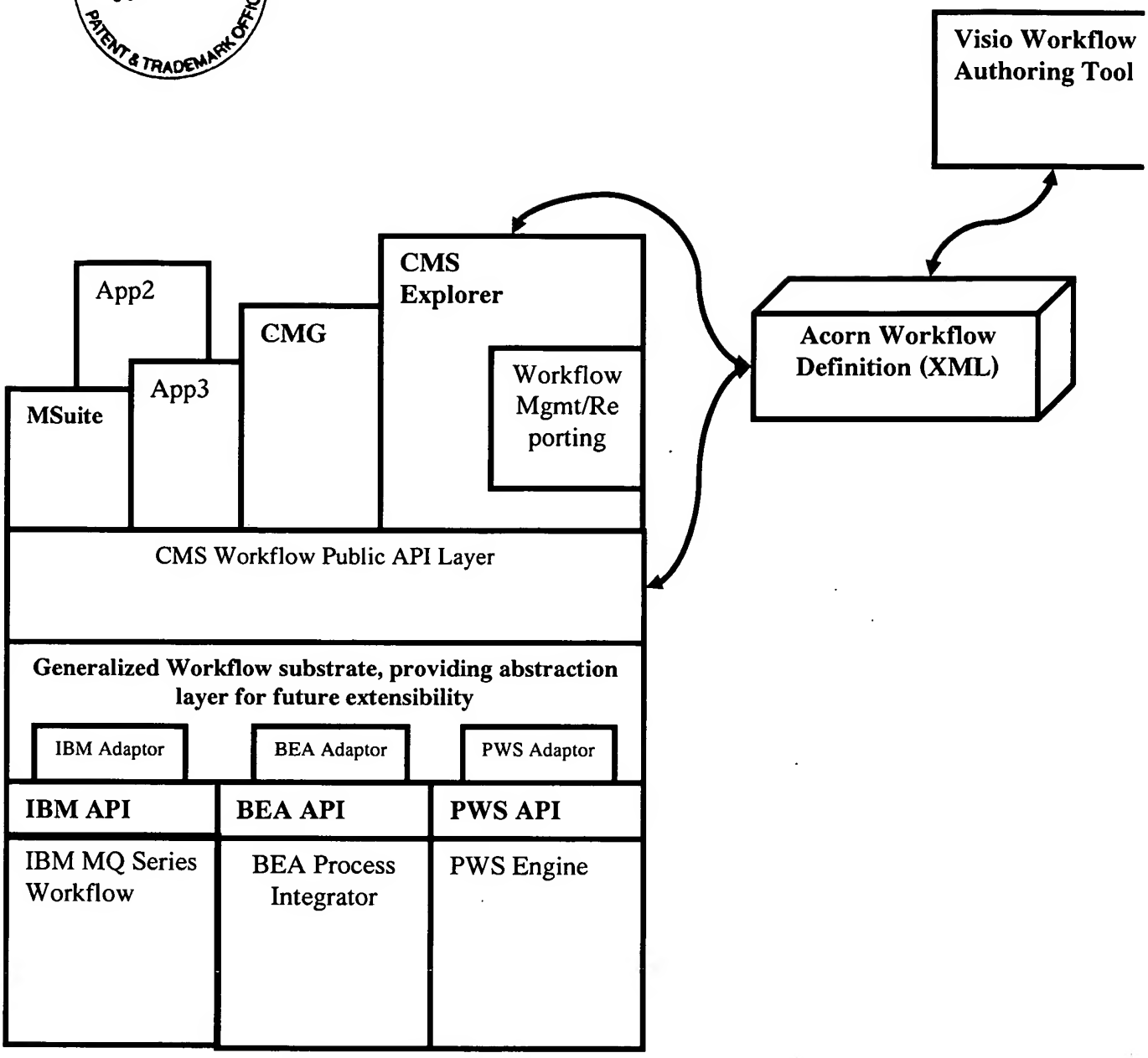
Workflow transition expressions must also support syntax for accessing portions of workflow-relevant data (see the glossary in http://www.wfmc.org/standards/docs/TC-1011_term_glossary_v3.pdf), since the modifications of workflow-relevant data may occur dynamically within a runtime process instance and may affect logical flow through that process.

The construction of this syntax must be craft to specifically support the use of XSL and XSLT (http://www.w3.org/Style/XSL/) for transforming the workflow transition syntax to another syntax (the target syntax need not be XML).

EXHIBIT B

Section 1. Conceptual Description of New Component/Feature

The *CMS Workflow Infrastructure*, hereafter referred to as the *Infrastructure*, is a component within the Acorn CMS that provides workflow features to Vignette client applications. The figure below illustrates the conceptual architecture for the Infrastructure, its major components, and its relationship to client applications and pluggable workflow engines.



The Infrastructure provides a common interface to Vignette client applications for workflow services, with an extensible, pluggable framework for incorporation of Vignette and other thirdparty process engines to leverage those services. Logically, the Infrastructure is divided into two layers:

- EJB Public API
- Dependent Convergence Adapter

The Public *Application Programming Interface* (*API*) layer consists of a collection of *Enterprise Java Beans* (*EJB*) used by clients to access the workflow subsystem. This layer is generic with respect to the underlying process engine. The Dependent Convergence Adapter layer, hereafter referred to as the *Adapter*, consists of software that

adapts the generic EJB layer to the actual process engine. This layer encapsulates dependencies to various process engines, mapping generic services and objects in the EJB layer to the native API for each engine. The following sections outline design objectives and constraints for these layers.

The EJB Layer

The public API provides EJBs with methods to access the workflow subsystem. The object model is derived from the Joint Workflow Management Facility Model [1], hereafter referred to as the *Object Management Group (OMG)* model. Adherence to the OMG model in desired to market enterprise standards compliance and to expedite Infrastructure development by adopting an existing model. The latter is higher priority, and if the model unexpectedly proves a hindrance, it is not strictly required.

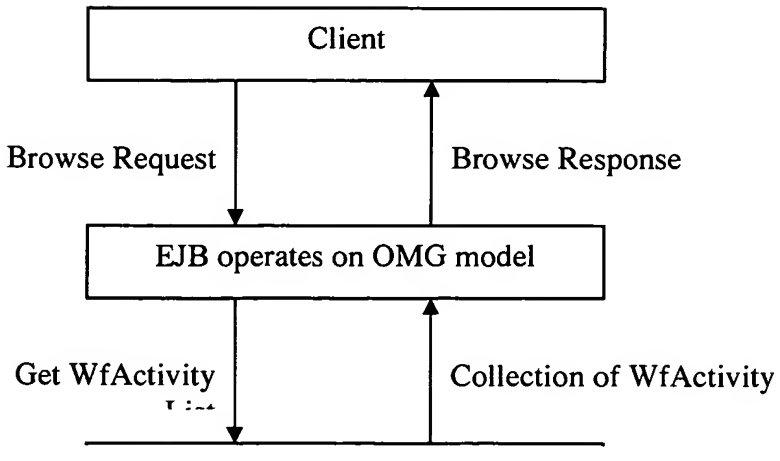
Regarding CORBA

Although that OMG model is defined using the *Common Object Request Broker Architecture (CORBA) Interface Definition Language (IDL)*, our EJB object model does not necessarily require the use of CORBA remote method invocation; we may elect to ignore the client-side stubs and server proxies, and directly implement the system-side Java stubs (interfaces).

Regarding Persistence

Note that it is the responsibility of the process engine, rather than the Infrastructure, to create and persist, as necessary, objects corresponding to process activities, resources, assignments, and audit events. Various process engine object models may differ, and none necessarily complies to the OMG model. Thus, while the OMG model consists of classes corresponding to activities, etc., instances of them within the Infrastructure are not persistent; in general, the Infrastructure creates transient OMG based objects in the course of processing a client workflow request. (This is not to say that the Infrastructure has no persistent objects, only that OMG objects reflecting those in the process engine are not.)

For instance, to satisfy a request to browse a worklist, the Infrastructure creates a collection of WfActivity objects satisfying the browse conditions, and uses these to generate the response. The actual collection is computed dynamically by interrogating the process engine. The native result it yields is converted by the adapter layer into the common, OMG representation used by the EJB layter. Figure 2 illustrates.



As shown in Figure 2, the business logic associated with servicing client workflow request resides in the EJB layer. It operates on OMG objects. These objects are transient reflections of persistent objects within the process engine. It is the role of the adapter layer to conform the EJB layer queries to the process engine API and translate the native result to the OMG model.

EXHIBIT C

Diana Schaffer

From: White, Eric [ewhite@vignette.com]
Sent: Monday, July 30, 2001 10:42 AM
To: 'jadair@graycary.com'
Subject: Vignette provisional Workflow patents

John,

We're working on a much more detailed description of the Extensible Workflow Architecture. We have a lot of knowledge about this one, because we've built it, so much of it is just documenting our detailed design.

OTOH, the Workflow Transition Syntax is not done and is mostly theory at this point, so I'm struggling with how to proceed with this one. Can you look over what I sent last week and provide me with some directed questions to help me focus?